PATENT APPLICATION
ATTORNEY DOCKET NO. 15115.020001

APPLICATION

FOR

UNITED STATES LETTERS PATENT

TITLE: **PROGRAM UPDATE APPARATUS AND METHOD**

APPLICANT: **Kenichi MURAI and Mitsuaki HIRONO**

"EXPRESS MAIL" Mailing Label Number: EV049244269US
Date of Deposit: March 5, 2002

22511
PATENT TRADEMARK OFFICE

PROGRAM UPDATE APPARATUS AND METHOD

Background of Invention

[0001] 本発明は、プログラム更新装置およびプログラム更新方法に関する。特に、本発明は、機器組込みシステムのバグフィックスやバージョンアップを行なう際に用いられるプログラム更新装置およびその更新方法に関するものである。

[0002] 機器組込みシステム等のプログラムでは、バグを解消するためのバグフィックスや、プログラムの改良、データの更新などに伴うバージョンアップが必要となることがある。そのため、従来からも機器組込みシステム等のプログラムの書き換えが行なわれることがある。

[0003] 図１はエネルギー使用量を計測する家庭用の使用量計測器の場合を表している。使用量計測器１は家屋の外壁等に設置されており、家庭２で使用される機器３には、第１のエネルギー供給ライン４や第２のエネルギー供給ライン５等を通じ、使用量計測器１を経由してエネルギーが供給されている。例えば、第１のエネルギー供給ライン４を通じて供給されるエネルギーとは電気のことであり、使用量計測器１はその電気使用量を計測し、積算している。また、第２のエネルギー供給ライン５を通じて供給されるエネルギーとはガスのことであり、使用量計測器１はそのガス使用量を計測し、積算している。

[0004] このような使用量計測器の目的は、電気やガス等のエネルギーの使用量を計測することであるが、これらのデータは料金徴収の目的以外にも使用されている。例えば、エネルギー供給元にとっては、１日ないし１週間の各時間ごとの細かい消費傾向や季節ごとの消費傾向を知ることができれば、そ

のデータに基づき、エネルギー消費者に対して各時間毎の細かな料金設定を行なうことが可能になり、サービスの向上を図ることができる。また、使用ピークの時間帯には高いエネルギー使用料金を設定することで、エネルギーの総消費量を抑制し、エネルギー消費が特定の時間帯に集中してエネルギーの供給不足になるのを防止することができる。

[0005] 上記のように料金徴収やデータ活用のためには、エネルギー供給元は、各家庭の使用量計測器を検針してエネルギー使用量を調べる必要がある。そのため、管理センタ（電気会社やガス会社等のエネルギー供給元）から派遣される計測人（検針人）が各家庭を訪問し、使用量計測器のメータを検針している。しかし、このようにして人手作業によって各家庭の使用量計測器を検針するという方法では、１日に検針できる戸数が限られており、人件費もかさみ、非常に効率が悪かった。

[0006] このようなことから、家庭に設置された使用量計測器を遠隔地から自動検針できるようにしたものがある。このような従来例では、図２に示すように、電柱８の上などにデータ収集器９を設置しておき、データ収集器９を中継して各家庭２の使用量計測器１を管理センタ６につないでいる。データ収集器９とは、一定時間毎に各家庭２の使用量計測器１からデータを収集し、そのデータを管理センタ６へ送る機能を持った機器である。データ収集器９は、電源の供給に困らない、電波的に遮断物が少ない、人に触れて欲しくないといったことから、電柱８の上など人の行きにくいところに設置されている。各家庭２の使用量計測器１や管理センタ６とデータ収集器９との間は、通信回線１０、１１などによって結ばれる。

[0007] このような方式によれば、多くの家庭の使用量計測器を遠隔地（管理センタ６）から瞬時に自動点検することができ、人件費の削減を図れ、また

１日に検針できる件数も制限が無くなり、効率的に検針を行なうことができるようになっている。

[0008] また、別な例として、家庭内に設置されている機器を外部から操作する場合を考える。図３に示す例では、家庭２内に設置されている機器１２Ａ、１２Ｂ、...と機器制御器１４とを信号線１３を通じて結んでおき、電話回線やネットワーク等の通信回線１５に接続された機器制御器１４を経由して各機器１２Ａ、１２Ｂ、...を家庭２の外から操作したり、機器１２Ａ、１２Ｂ、...からの情報を外部で取得できるようにしている。例えば、機器１２Ａが監視カメラであれば、電話回線に接続したモバイルコンピュータや携帯電話により外部から監視カメラの画像をモニターしたり、監視カメラの撮影方向を操作したりすることができ、機器１２Ｂがビデオレコーダであれば、ビデオレコーダの録画予約を外部から行なったり、機器１２Ｃがホームセキュリティに必要な監視センサであれば、遠隔地から外部にいてセキュリティシステムからの通知を取得することができる。

[0009] 機器制御器１４に接続される機器にはさまざまなものが想定され、機器ごとに持っている情報が違うため、機器制御器１４に機器を接続するにあたっては、その機器に応じて機器制御器１４のシステムプログラムを書き換える必要がある。

[0010] また、ホームセキュリティに関連するシステムで用いられている場合には、ホームセキュリティ動作中では機器制御器１４がフリーズすることは許されない。フリーズするとは、機器の応答が無くなり、外部へ状態を通知できなくなったり、外部からの入力を受け付けない状態になることである。ホームセキュリティとは、不審者の侵入、火災、ガス漏れ、非常通報、緊急・救急の検知をするものであり、機器制御器１４がフリーズすると、ホー

ムセキュリティが意味をなさず、機器制御器１４の信頼性が損なわれる問題がある。

[0011] 通信回線を通じて外部と結ばれた機器組込みシステム、例えば上記のようなデータ収集器や機器制御器における技術的課題を説明する。

[0012] データ収集器９や機器制御器１４においては、システム（動作プログラム）にバグが見つかることがあり、そのような場合にはシステムプログラムのバグフィックスが必要になる。また、データ収集器９や機器制御器１４に新しい機器（例えば、使用量計測器１など）を追加接続した場合や、データ収集器９や機器制御器１４に新しい機能を追加したい場合には、それぞれのシステムプログラムにも対応する機能を追加したり、システムプログラムをアップデート（更新）したりする必要がある。そして、バグフィックスや機能追加のためにプログラムをアップデートするためには、データ収集器９や機器制御器１４に組み込まれているシステムのプログラムを書き換える必要がある。

[0013] このようなプログラムの書き換えが必要となる場合、例えばアップデート用のプログラムを提供してユーザーサイドで書き換えをして貰うということになると、機械に苦手なユーザーではプログラムの書き換えやフリーズした際の復旧作業を行なって貰うことは期待できない。また、個々のデータ収集器９や機器制御器１４はプログラム提供者（管理センタなど）から遠隔地にあることが多く、しかも、データ収集器９や機器制御器１４では人の行きにくい場所（電柱の上など）に設置されていることが多い。よって、システムプログラムの書き換えを行なう場合には、人の手を介さず、通信回線等を通じて、自動的に書き換えを行なう必要がある。

[0014] さらに、ホームセキュリティに関連した機器制御器１４等では、書き換えのためにシステムを停止させることはできず、システムを稼働させた状

況下で書き換えを行なう必要があり、書き換え中にシステムがフリーズしたような場合でも、とりあえず再起動させてシステムを稼働させる必要がある。

[0015] よって、機器組込みシステムによっては、通信回線等を通じて自動的にプログラムの書き換えを行えると共に万一の場合にもシステムがフリーズすることのないアップデート方法が求められている。

[0016] 従来でも、通信回線を通じて機器組込みシステムのプログラムを書き換える処理は行なわれている。しかし、それは、正しく動くプログラムが前提で、それが正しく書き込めたか否かをチェックするだけであった。そのため、致命的な部分は書き換えないようにしている。また、プログラムのチェックにチェックサムのみを利用している場合には、複数のエラーが発生した場合、チェックする値が偶然正しい値に一致する可能性があり、正しく書き込めたと誤認識する恐れがあった。

[0017] さらに、従来の書き換え方式では、バグのあるプログラムには対応できず、オペレーティングシステムが動作しなければ、ネットワークダウンロードなど複雑な処理は行えなかった。

Summary of Invention

[0018] 本発明の目的とするところは、機器組込みシステム（特に、常時動作する必要のある機器のシステム）において、自動的にシステムのアップデートを行え、しかもシステムをフリーズさせることのないプログラム更新装置およびその更新方法を提供することにある。

[0019] 本発明にかかるプログラム更新装置は、メモリ機器又は通信回線を通じて動作プログラムをアップデートするためのプログラム更新装置であって、動作プログラムが格納されている第１の記憶領域とは別の第２の記憶領域を確保する手段と、第１の記憶領域に格納されている第１の動作プログラムと

は別の第２の動作プログラムを第２の記憶領域に格納する手段と、第１の動作プログラム又は第２の動作プログラムのいずれかを選択的に起動させる選択起動手段と、起動を選択された動作プログラム又は当該動作プログラムの上で動作するアプリケーションプログラムの起動の失敗を検知する失敗検知手段と、前記アプリケーションプログラムの動作に必要な設定を変更する修復アプリケーションとを備え、前記失敗検知手段が起動を選択された動作プログラムの起動の失敗を検知した場合には、別の動作プログラムを選択して起動させ、前記失敗検知手段が前記アプリケーションプログラムの起動の失敗を検知した場合には、修復アプリケーションにより、前記アプリケーションプログラムの動作に必要な設定を修復させるようにしたことを特徴としている。ここで、動作プログラムとは、機器とアプリケーションを制御するプログラムであって、オペレーティングシステム（ＯＳ）も含まれる。また、メモリ機器とは、プログラム更新装置に接続することのできるメモリであれば特に限定されるものでなく、メモリカード、メモリスティックなどがある。通信回線は、有線でも無線でもよく、例えばインターネットや携帯電話、ＰＨＳ（Personal Handy-phone System）によるネットワーク（通信網）を用いることができる。

[0020] このプログラム更新装置にあっては、第１及び第２の記憶領域のうち一方の記憶領域にそれまで動作していた動作プログラムを残したままで、他方の記憶領域に新しい動作プログラムをダウンロードし、第１の記憶領域に格納されている動作プログラムと第２の記憶領域に格納されている動作プログラムのうち、いずれかを選択的に起動させることができるので、いずれか一方の記憶領域に新しい動作プログラムをダウンロードした後、その動作プログラムの起動に失敗した場合には、他方の記憶領域に保存されている以前の動作プログラムを再起動させることがき、動作プログラムのアップデートに失敗しても動作プログラムを組み込まれている機器がフリーズするのを回

避できる。さらに、動作プログラムが起動したとしても、新しい動作プログラムとアプリケーションプログラムの相性が悪くアプリケーションプログラムが動作しないときであっても、アプリケーションが動作していないことを検出し、アプリケーションプログラムを動作させるための設定を修復アプリケーションが行い、アプリケーションプログラムを動作させることができる。また、このプログラム更新装置にあっては、通信回線を介して遠隔から自動的にアップデートを行なわせることができる。従って、このプログラム更新装置によれば、遠隔から自動的に動作プログラムをアップデートさせることができ、しかもアップデートに失敗した場合でもシステムがフリーズするのを防止することができる。

[0021] また、前記プログラム更新装置においては、プログラム更新装置の外部に通知する通知手段をさらに備え、前記動作プログラムのアップデートが正常になされなかった場合には、前記通知手段により、アップデートが正常になされなかった情報を前記プログラム更新装置の外部に通知させるようにすることが望ましい。ここで通知手段とは、通信回線を使って出力する手段であっても良いし、その旨を知らせる表示装置、またはその両方であってもよい。このような実施形態によれば、アップデートに失敗して動作プログラムが更新されずに動作を続けている場合に、プログラム更新装置からアップデートのデータを送る側に通知することにより、再度アップデート作業を試みることが可能となる。

Brief Description of Drawings

[0022] 図１はエネルギー使用量を計測する家庭用の使用量計測器の使用状態を示す図である。

[0023] 図２は家庭に設置された使用量計測器を遠隔地から自動検針できるようにした構成を説明する図である。

[0024] 図３は家庭内に設置されている機器を外部から操作するための機器制御器を説明する図である。

[0025] 図４は本発明に係るプログラム更新方式の実行手順を示すフロー図である。

[0026] 図５は図４の復旧起動ステップの詳細を示すフロー図である。

[0027] 図６は図４のプログラム更新方式を具体的に説明する図である。

[0028] 図７は本発明に係るプログラム更新方式を実行するためのシステム構成を示す図である。

[0029] 図８はブートステータスレジスタの構造を示す図である。

[0030] 図９は図７に示したシステムを機能として表現した図である。

[0031] 図１０は図７～図９のシステム構成におけるアップデート方法の手順を示す図である。

[0032] 図１１はＯＳ上でＪａｖａＶＭが動いているＪａｖａ実行環境の一例を示す図である。

[0033] 図１２は動作プログラムをダウンロードしてシステムをアップグレードする際の全体の流れを表した図である。

[0034] 図１３は図１２の続図である。

[0035] 図１４は正常解決機能における起動プログラム決定部の処理を説明する図である。

[0036] 図１５は復旧挑戦機能における起動プログラム決定部の処理を説明する図である。

[0037] 図１６は確実復旧機能における起動プログラム決定部の処理を説明する図である。

[0038] 図１７は正常解決機能及び復旧挑戦機能における起動アプリケーション決定部の処理を説明する図である。

[0039] 図１８は確実復旧機能における起動アプリケーション決定部の処理を説明する図である。

[0040] 図１９は正常解決機能、復旧挑戦機能、確実復旧機能におけるアプリケーション動作監視部の処理を説明する図である。

[0041] 図２０は正常解決機能におけるソフトウエアの流れを示すフロー図である。

[0042] 図２１は復旧挑戦機能におけるソフトウエアの流れを示す図である。

[0043] 図２２は第１の確実復旧機能におけるソフトウエアの流れを示す図である。

[0044] 図２３は第２の確実復旧機能におけるソフトウエアの流れを示す図である。

[0045] 図２４は自動検針システム（データ収集器）のモジュール構成図である。

[0046] 図２５は図２４の自動検針システムのハードウエア構成図である。

[0047] 図２６は図２５のＲＯＭとコンパクトフラッシュメモリの状態を示す図である。

Detailed Description

[0048] 本発明に係るプログラム更新方式は、図４に示すように大まかには、ダウンロードステップ（ステップＳ２２）、新起動ステップ（ステップＳ２３）、復旧起動ステップ（ステップＳ２５）からなる。そして、システムのアップデートが開始すると（ステップＳ２１）、システムに新動作プログラム（アップデートされた動作プログラム）をダウンロードし、システム内のプログラム動作用とは別領域にある記憶装置空き領域（例えば、複数のＲＯＭ領域において、旧動作プログラムが稼働中でないＲＯＭ領域）に新動作プログラムを転送し、書き込みを行なう（ダウンロードステップ、ステップＳ２２）。新動作プログラムがダウンロードされて転送と書き込みが終了したら、ダウンロードの時点で動作していた旧動作プログラムに代えてダウンロードした新動作プログラムを起動させる（新起動ステップ、ステップＳ２３）。そして、新動作プログラムが正常に起動したか否かを判定し（ステップＳ２４）、新動作プログラムが正常に起動していたら、正常解決機能によってアップデートが完了したと判断し、アップデート作業を完了する（ステップＳ２６）。

[0049] これに対し、ステップＳ２４において、新動作プログラムが正常に起動していないと判断された場合には、復旧挑戦機能および確実復旧機能を内容とするシステム復旧動作を行なってシステムを再起動させてフリーズから復旧させ（復旧起動ステップ、ステップＳ２５）、アップデート作業を完了する（ステップＳ２６）。このとき、新動作プログラムにアップデートされている場合と、確実に動作させるために旧動作プログラムに戻っている場合がある。旧動作プログラムのまま動作する場合には、プログラム更新装置の外部に通信回線又は表示等によって通知し、次の新動作プログラムのアップデート作業を待つ。

[0050] ここで上記正常解決機能とは、ダウンロードした新動作プログラムが正常起動し、システムの動作プログラムが旧動作プログラムから新動作プログラムに切り換わってシステムが新動作プログラムで動作するようにする機能であり、復旧挑戦機能および確実復旧機能とは、ダウンロードした新動作プログラム等が正常起動できない場合に、システムがフリーズしないよう復旧させるための機能である。

[0051] しかして、本発明に係るプログラム更新方式では、旧動作プログラムが稼働しているシステムに新動作プログラムをダウンロードし、稼働するプログラムを旧動作プログラムから新動作プログラムに切り替えて新動作プログラムを実行させる。そして、フリーズ等により新動作プログラムの起動に失敗した場合には、もとの旧動作プログラムを再起動し、旧プログラムを実行させる。あるいは、旧動作プログラム又は新動作プログラムが稼働している状態で、実行アプリケーションがフリーズ等により実行停止した場合には、修復アプリケーションを実行して実行アプリケーションを修復させるものである。ここで、動作プログラムとは機器とアプリケーションを制御するプログラムであって、オペレーティングシステム（ＯＳ）も含まれる。また、旧動作プログラムとは、ダウンロード実行時に稼働中の動作プログラムであり、新動作プログラムとは、置換用の動作プログラム（アップデートプログラム）である。実行アプリケーションとは、動作プログラム上で動く、機器の利用目的を実行するアプリケーションプログラムである。修復アプリケーションとは、動作プログラム上で動く、修復機能を持った必要最低限のアプリケーションプログラムであり、動作プログラム上でアプリケーションプログラムを動作させるための設定を行う。このアップデート作業においては、遠隔からプログラムのアップデートを行なうため、通信回線を通じてプログラムをダウンロードする。通信回線は、有線でも無線でもよく、例えばインタ

ーネットや携帯電話、ＰＨＳによるネットワーク（通信網）を用いることができる。

[0052] なお、システムのフリーズは、(1)ダウンロードした動作プログラムが破壊されている場合、(2)初期化データなどのデータが破壊されている場合、あるいは、(3)動作プログラムは正常に動作していても実行アプリケーションが正常に起動しない場合に起きるが、本発明のアップデート方式では、以下に説明するようにしてシステムのフリーズを回避する。

[0053] 図５はシステム復旧動作（復旧起動ステップ、ステップＳ２５）の詳細なフローである。新動作プログラムが正常起動しない場合には、復旧起動ステップを開始する（ステップＳ３１）。まず新動作プログラムが正常動作しているかどうか判断し（ステップＳ３２）、新動作プログラムが正常動作している場合には、実行アプリケーションが動作していないので、第１の確実復旧機能で修復アプリケーションを起動させ、実行アプリケーションを実行可能な状態に修復し（ステップＳ３３）、終了する（ステップＳ３８）。ステップＳ３２で新動作プログラムが正常動作していなければ、復旧挑戦機能で旧動作プログラムからの立ち上げを行う（ステップＳ３４）。旧動作プログラムは、このアップデート動作まで動いていたプログラムであるから、間違いなく起動する。その旧動作プログラムの上で実行アプリケーションが起動するか判断し（ステップＳ３５）、実行アプリケーションが起動すれば、旧動作プログラムのままで動作している旨の通知を出し（ステップＳ３７）、終了する（ステップＳ３８）。旧動作プログラムの上で実行アプリケーションが起動しない場合には、新動作プログラムによって起動のための設定が変更されている可能性があり、第２の確実復旧機能で修復アプリケーションを起動させ、実行アプリケーションを実行可能な状態に設定を修復し（ステッ

プＳ３６）、旧動作プログラムのままで動作している旨の通知を出し（ステップＳ３７）、終了する（ステップＳ３８）。

[0054] 図６は上記方式を具体的に説明する図である。図６（ａ）はダウンロード前の状態を表しており、システム内では旧動作プログラム２１により実行アプリケーション２２が動作している。なお、図６において、符号「ＲＵＮ」は動作中のアプリケーションを表している。ついで、アップデート作業が開始すると、図６（ｂ）に示すように、ネットワーク等の通信回線又はメモリーカード等を通じてシステムにデータ２５がダウンロードされ、図６（ｃ）に示すように新動作プログラム２４が空き領域（例えば、複数のＲＯＭ領域ＲＯＭ１とＲＯＭ２があり、ＲＯＭ１の領域が稼働中であるとすれば、ＲＯＭ２の領域）に転送され、そこに書き込まれる。新動作プログラム２４の書き込みが終了すると、図６（ｄ）に示すように、正常解決機能によって書き込まれた領域（例えば、上記ＲＯＭ２の領域）から新動作プログラム２４が起動され、動作プログラムが旧動作プログラム２１から新動作プログラム２４に切り替えられ、実行アプリケーション２２が新動作プログラム２４の上で実行され、修復アプリケーション２３も新動作プログラム２４の上で実行可能となる。

[0055] 新動作プログラム２４を起動すると（新起動ステップ）、新動作プログラム２４の起動に成功（ＯＫ）したか、失敗（ＮＧ）したか判断し、新動作プログラム２４のダウンロードが正常完了していて新動作プログラム２４の起動に成功していれば（図６（ｅ））、アップデート機能が完了する（図６（ｆ））。

[0056] このようにして正常解決機能が働いてダウンロードした新動作プログラム２４でシステムが正常に起動されると、機器の目的を妨げることなく（すなわち、システムを停止させることなく）システムの書き換えが完了し、

機器の目的を実行しつつ、同時にシステムの書き換えを行なうという本発明の目的が達成される。

[0057] これに対し、新動作プログラム２４の起動に失敗した場合には、失敗の状況に応じて復旧挑戦機能、第１又は第２の確実復旧機能の３種類のシステム復旧機能のうちから最適な方法を実行する。すなわち、復旧挑戦機能とは、正常解決機能による新動作プログラム２４の起動に失敗した場合に、図６（ｇ）に示すように、ダウンロードした新動作プログラム２４を実行せず、元の旧動作プログラム２１を再起動して旧動作プログラム２１でシステムを実行し直すものである。また、第１の確実復旧機能とは、正常解決機能により新動作プログラム２４が正常に起動されても、実行アプリケーション２２の起動に失敗した場合、図６（ｈ）に示すように、ダウンロードした新動作プログラム２４が起動された状況において、修復アプリケーション２３を実行して実行アプリケーション２２を修復するものである。第２の確実復旧機能とは、正常解決機能による新動作プログラム２４の起動に失敗した後、復旧挑戦機能によって旧動作プログラム２１が復旧されたが、実行アプリケーション２２の起動に失敗した場合に、図６（ｉ）に示すように、旧動作プログラム２１が実行されている状況において、修復アプリケーション２３を実行して実行アプリケーション２２を修復するものである。ここで修復アプリケーションは、旧動作プログラム２１又は新動作プログラム２４の上で実行アプリケーション２２が動作できるように設定を調整する。

[0058] 新動作プログラム２４の起動失敗の状況が、新動作プログラム２４は正常に起動したが、実行アプリケーション２２がフリーズした場合には、第１の確実復旧機能を働かせて実行アプリケーション２２を修復し（図６（ｈ））、新動作プログラム２４の上で実行アプリケーション２２を動作させ、アップデート機能が完了する（図６（ｆ））。また、新動作プログラム

２４がフリーズした場合には、復旧挑戦機能を働かせて動作プログラムを新動作プログラム２４から旧動作プログラム２１に戻して旧動作プログラム２１を再起動し（図６（ｇ））、システムが正常に動くようになれば修復を完了し（図６（ｊ））、旧動作プログラム２１の上で実行アプリケーション２２を動作させ、アップデート機能が完了する（図６（ｆ））。もし、復旧挑戦機能を作動させてもシステムが復旧されなかった場合には、すなわち旧動作プログラム２１は正常に再起動したが、実行アプリケーション２２がフリーズした場合には、第２の確実復旧機能を働かせて旧動作プログラム２１の上で修復アプリケーション２３を実行して実行アプリケーション２２を修復し（図６（ｉ））、旧動作プログラム２１の上で実行アプリケーション２２を動作させ、アップデート機能が完了する（図６（ｆ））。

[0059] このようにして正常解決機能が失敗した場合でも、復旧挑戦機能によって旧動作プログラムでシステムを再起動させることができ、あるいは確実復旧機能によって修復アプリケーションで実行アプリケーションを修復することができるので、決してシステムがフリーズすることが無く、新動作プログラムの起動に失敗してもシステムをフリーズさせないという本発明の目的が達成される。なお、アップデート機能がフリーズすることなく完了するが、ここで完全に新動作プログラムの置き換えの目的を達成しているのは図６（ｄ）（ｈ）である。図６（ｇ）（ｉ）は旧実行プログラムで動作し続けており、この場合プログラム更新装置は新実行プログラムのダウンロード元に、新動作プログラムが動作していない旨の通知を出し、再度ダウンロードを要求しても良いし、次の新実行プログラムのダウンロードを待っても良い。

[0060] 次に、上記のようなリモートプログラム更新方式を実行するためのシステム構成を図７に示す。このシステムはデータ収集器や機器制御器等の機器に組み込まれており、マイクロプロセッサ（ＣＰＵ）２６、パワーオンリ

セット（Power On Reset）回路２７、ウォッチドックタイマ（ＷＤＴ）２８、ブートステータスレジスタ２９、ブートプログラムＲＯＭ３０、プログラムＲＯＭ３１、３２、ＲＡＭ３３、入出力部（Ｉ／Ｏ）３４、ネットワーク３５、メモリカード３６等によって構成され、これらはバスライン３７を通じて結ばれている。ここで、マイクロプロセッサ２６は、システム全体を制御するものである。パワーオンリセット回路２７は、システムをオン、オフし、あるいはハードウエアリセットを掛けるための回路である。ウォッチドックタイマ２８は、システムがタイムアウトした際にソフトウエアリセットを掛けることができるものである。ブートプログラムＲＯＭ３０は、最小限のブートプログラム領域である。プログラムＲＯＭ３１、３２は、２重化されたプログラム格納領域である。ブートプログラムＲＯＭ３０及びプログラムＲＯＭ３１、３２は、同一メモリの領域を分割したものであってもよい。ＲＡＭ３３は、正常解決機能等を実行する際など、一時的にデータを保存するものである。ネットワーク３５は、インターネット等を通じて外部（公衆回線等）と結ばれており、新動作プログラムはネットワーク３５を通じていずれか一方のプログラムＲＯＭ３１、３２にダウンロードされる。また、新動作プログラム２４は、入出力部３４やメモリカード３６からインストールできるようにしてもよい。

[0061] ブートステータスレジスタ２９は、起動させた動作プログラムや実行アプリケーションが異常となったかどうかを記憶しておくレジスタであって、ソフトウエアリセットによっては値が変更されない。ブートステータスレジスタ２９は、図８に示すように、最下位ビットにＢＯＯＴフラグを持ち、その上位ビットにＣＯＮＦＩＧフラグを持っている。ＢＯＯＴフラグは、電源オンの直後には”０”となっているが、初回ブートイメージ異常があると値”１”が保存される。また、ＣＯＮＦＩＧフラグは、電源オンの直後には”０”となっているが、初期設定値に異常があると値”１”が保存される。システムは、

新動作プログラム２４を起動した際、新動作プログラム２４の異常でソフトウエアリセットがかかると、ブートステータスレジスタ２９のＢＯＯＴフラグをチェックし、旧動作プログラム２１が格納されているＲＯＭ領域と新動作プログラム２４が格納されているＲＯＭ領域のうちいずれのＲＯＭ領域から動作プログラムを起動するかを決定する。また、実行アプリケーション２２の異常でソフトウエアリセットがかかると、システムは、ブートステータスレジスタ２９のＣＯＮＦｌＧフラグをチェックし、実行アプリケーション２２と修復アプリケーション２３のうちどちらのアプリケーションを起動するか決定する。

[0062] 図９は図７に示したシステムが各プログラムによって動作する機能を表現したものである。これらの機能は、

（ｆ１）ダウンロードされた新動作プログラム２４の格納場所（メモリ領域）を決定する機能、

（ｆ２）決定された格納場所へダウンロードされた新動作プログラム２４を格納する機能、

（ｆ３）格納された新動作プログラム２４をチェックする機能、

（ｆ４）格納された新動作プログラム２４を起動する機能、

（ｆ５）ダウンロードする前に動作していた旧動作プログラム２１を起動する機能、

（ｆ６）起動された新動作プログラム２４または旧動作プログラム２１の動作をチェックする機能、

（ｆ７）チェックされた動作プログラム上で実行アプリケーション２２を起動する機能、

（ｆ８）チェックされた動作プログラム上で修復アプリケーション２３を起動する機能、

（ｆ９）起動された実行アプリケーション２２または修復アプリケーショ

ン２３の動作をチェックする機能、

から構成されており、新動作プログラム２４のダウンロードから始まって図９の矢印の順に実行され、アップデート機能の完了によって終了する。

[0063] しかして、図７～図９のようなシステム構成において、本発明のリモートプログラム更新方式は図１０に示す手順に沿って実行される。以下、図９の機能を参照しながら、図５に示したフローを詳細に、図１０に沿ってアップデートの手順を説明する。なお、図５において示した、旧動作プログラムのままである旨の通知（ステップＳ３７）は付加機能であるため、図１０では省略してある。新動作プログラム２４がネットワーク３５を通じて外部からダウンロードされ、いずれかのＲＯＭに格納されると（ステップＳ４１、機能ｆ１、ｆ２）、システムは新動作プログラムが正常に起動したかどうかチェックする（ステップＳ４２）。すなわち、新動作プログラム２４がダウンロードされると、システムは、格納された新動作プログラム２４をチェックし（機能ｆ３）、異常がなければする新動作プログラム２４を起動し（機能ｆ４）、さらに、ブートステータスレジスタ２９をチェックすることにより、起動された新動作プログラム２４が正常に起動したかどうかチェックする（機能ｆ６）。

[0064] 新動作プログラム２４が正常に起動している場合には、起動された新動作プログラム２４上で実行アプリケーション２２を起動させ（機能ｆ７）、実行アプリケーションが正常に起動したかどうかチェックする（ステップＳ４３）。すなわち、システムは、ブートステータスレジスタ２９をチェックすることによって実行アプリケーションに異常がないかチェックする（機能ｆ９）。そして、実行アプリケーションが正常に起動していれば、システムの起動を完了する（アップデート完了、ステップＳ４５）。

[0065] これに対し、ステップＳ４３で実行アプリケーション２２に異常があった場合には、ブートステータスレジスタ２９のＣＯＮＦＩＧフラグが”１”になっているので、ウォッチドックタイマ２８のオーバータイムによってソフトウエアリセットが掛かり、ブートステータスレジスタ２９のＣＯＮＦＩＧフラグがチェックされると、修復アプリケーション２３が起動される。起動された修復アプリケーション２３はシステム（実行アプリケーション２２）の設定を修復し（ステップＳ４４、機能ｆ８）、システムの起動を完了する（ステップＳ４５）。

[0066] また、ステップＳ４２で新動作プログラム２４が異常である場合には、ブートステータスレジスタ２９のＢＯＯＴフラグが”１”になっているので、ウォッチドックタイマ２８のオーバータイムによってソフトウエアリセットが掛かり、ブートステータスレジスタ２９のＢＯＯＴフラグがチェックされると、旧動作プログラム２１でシステムを再起動してシステムを旧動作プログラム２１に復旧させる（ステップＳ４６、機能ｆ５）。ついで、旧動作プログラム２１上で実行アプリケーション２２を起動させ（機能ｆ７）、実行アプリケーション２２が正常に起動したかどうかチェックする（ステップＳ４７）。そして、実行アプリケーション２２が正常に起動していれば、システムの起動を完了する（アップデート完了、ステップＳ４５）。これに対し、実行アプリケーション２２が異常である場合には、ウォッチドックタイマ２８のオーバータイムによってソフトウエアリセットが掛かり、ブートステータスレジスタ２９のＣＯＮＦＩＧフラグがチェックされると、修復アプリケーション２３を起動してシステム（実行アプリケーション２２）の設定を修復し（ステップＳ４８、機能ｆ８）、システムの起動を完了する（ステップＳ４５）。

[0067] 次に、ＯＳ（オペレーティングシステム）上でＪａｖａＶＭ（仮想マシン）が動いているＪａｖａ実行環境下においてアップデートが行なわれる様子を具体的に説明する。図１１はＯＳ上でＪａｖａＶＭが動いているＪａｖａ実行環境の一例を示しており、システム起動時に働くブートプログラム４１、システム全体を制御するＯＳ４２、ＯＳ４２の違いを吸収して共通のプラットフォームを提供するＪａｖａＶＭ４３、ＪａｖａＶＭ上で動くクラスライブラリ４４、機器の利用目的を実行する実行アプリケーション４６、必要最小限の修復機能を持った修復アプリケーション４７によって構成されている。ここで、ＯＳ４２、ＪａｖａＶＭ４３及びクラスライブラリ４４がアップデート対象となる動作プログラム４５である。以下説明するために、アップデート対象部分を動作プログラムに限定しているが、２重化したメモリで動作プログラムを切り替えるので、同じメモリ上にある実行アプリケーションもアップデートの対象になることは言うまでもない。

[0068] 一方、上記動作プログラム４５をアップデートするためのアップデート実行部は、新しいＪａｖａＶＭ及びＯＳと以前からのＪａｖａＶＭ及びＯＳのうちいずれの動作プログラムを起動させるかを決める起動プログラム決定部４８、どのアプリケーションプログラムを起動させるかを決定する起動アプリケーション決定部４９、起動したアプリケーションの動作を監視するアプリケーション動作監視部５０を備えている。

[0069] この環境下におけるアップデート方式でも、正常解決機能、復旧挑戦機能、第１及び第２の確実復旧機能が実現されているが、まず、これらの機能の詳細は省き、動作プログラムをダウンロードしてシステムをアップグレードする際の全体の流れを図１２及び図１３により説明する。このシステムでは、新しいＪａｖａＶＭ及びＯＳ（新動作プログラム）をネットワーク３５等を通じて外部からダウンロードすると（ステップＳ５１）、ダウンロー

ドされたＪａｖａＶＭ及びＯＳは、現在動作しているＪａｖａＶＭ及びＯＳ（旧動作プログラム）により使用されていないＲＯＭ領域に格納される（ステップＳ５２）。ついで、システムにハードウエアリセットまたはソフトウエアリセットを掛け、システムを再起動する（ステップＳ５３）。その際、システムの起動プログラム決定部４８は、ブートステータスレジスタ２９をチェックすることにより、新旧いずれのＪａｖａＶＭ及びＯＳを起動させるかを決定する（ステップＳ５４）。起動するＪａｖａＶＭ及びＯＳが決まると、システムは決定されたＪａｖａＶＭ及びＯＳが正常か否かをチェックサムを用いてチェックする（ステップＳ５５）。このチェックにより、決定されたＪａｖａＶＭ及びＯＳが正常でないと判断されると、起動するＪａｖａＶＭ及びＯＳを他方のＪａｖａＶＭ及びＯＳに変更する（ステップＳ５６）。

[0070] こうして正常に起動できるＪａｖａＶＭ及びＯＳが決定されると、ウォッチドックタイマ２８が起動し（ステップＳ５７）、ブートステータスレジスタ２９のＢＯＯＴフラグを反転させ（ステップＳ５８）、ＪａｖａＶＭ及びＯＳを起動する（ステップＳ５９）。

[0071] ついで、起動アプリケーション決定部４９が、起動させるアプリケーションを決定すると（ステップＳ６０）、決定したアプリケーションをロードし（ステップＳ６１）、ブートステータスレジスタ２９のＢＯＯＴフラグとＣＯＮＦＩＧフラグを反転させる（Ｓ６２）。これによって、ステップＳ５８で反転されたＢＯＯＴフラグが元に戻る。また、この段階に達するまでにＪａｖａＶＭ及びＯＳがフリーズしていれば、ウォッチドックタイマ２８のタイムアウトでソフトウエアリセットがかかっている筈であるから、この時点で、ＪａｖａＶＭ及びＯＳが正常に起動したと判断できる。従って、ウォッチドックタイマ２８を更新し（ステップＳ６３）、アプリケーション動

作監視部５０により、ロードしたアプリケーションの動作を監視する（ステップＳ６４）。

[0072] 図１４～図１９は、図１２及び図１３に示した処理中で行なわれる正常解決機能、復旧挑戦機能、確実復旧機能の処理手順を示している。なお、図１４～図１９において、共通部とは、正常解決機能、復旧挑戦機能、確実復旧機能に共通な図１２、１８の処理を指している。まず、図１４により正常解決機能における起動プログラム決定部４８の処理を説明する。図１２のステップＳ５８でブートステータスレジスタ２９のＢＯＯＴフラグが反転された後、起動プログラム決定部４８はブートステータスレジスタ２９のＢＯＯＴフラグをチェックする（ステップＳ７１）。そして、起動プログラム決定部４８は、ＢＯＯＴフラグが反転しているか否か調べ（ステップＳ７２）、反転していれば、ダウンロードする前のＪａｖａＶＭ＋ＯＳで起動するように決定し（ステップＳ７４）、ＢＯＯＴフラグが反転していなければダウンロードしたＪａｖａＶＭ＋ＯＳで起動するように決定する（ステップＳ７３）。ダウンロードしたＪａｖａＶＭ及びＯＳに異常がなくて正常解決機能が働く場合には、ＢＯＯＴフラグは反転していないので、起動プログラム決定部４８はダウンロードしたＪａｖａＶＭ及びＯＳを起動するように決定する（ステップＳ７３）。

[0073] また、図１５は復旧挑戦機能における起動プログラム決定部４８の処理を説明している。図１２のステップＳ５８でブートステータスレジスタ２９のＢＯＯＴフラグが反転された後、起動プログラム決定部４８はブートステータスレジスタ２９のＢＯＯＴフラグをチェックする（ステップＳ７１）。そして、起動プログラム決定部４８は、ＢＯＯＴフラグが反転しているか否か調べ（ステップＳ７２）、ＢＯＯＴフラグが反転していれば、ダウンロードする前のＪａｖａＶＭ＋ＯＳで起動するように決定し（ステップＳ７４）、

ＢＯＯＴフラグが反転していなければダウンロードしたＪａｖａＶＭ＋ＯＳで起動するように決定する（ステップＳ７３）。ダウンロードしたＪａｖａＶＭ及びＯＳに異常があって復旧挑戦機能が働く場合には、正常解決機能で動作中に、図１３のステップＳ６２に達する前にウォッチドックタイマ２８がタイムアウトしてソフトウエアリセットがかかり、ＢＯＯＴフラグは反転したままになるので、起動プログラム決定部４８はダウンロードする前のＪａｖａＶＭ及びＯＳを起動するように決定する（ステップＳ７４）。

[0074] また、図１６は確実復旧機能における起動プログラム決定部４８の処理を説明している。図１２のステップＳ５８でブートステータスレジスタ２９のＢＯＯＴフラグが反転された後、起動プログラム決定部４８はブートステータスレジスタ２９のＢＯＯＴフラグをチェックする（ステップＳ７５）。そして、起動プログラム決定部４８は、ＢＯＯＴフラグが反転しているか否か調べ（ステップＳ７６）、ＢＯＯＴフラグが反転していれば、直前に動作していないＪａｖａＶＭ＋ＯＳを起動するように決定し（ステップＳ７８）、ＢＯＯＴフラグが反転していなければ直前に動作していたＪａｖａＶＭ＋ＯＳで起動するように決定する（ステップＳ７７）。確実復旧機能が働く場合には、第１の確実復旧機能でも第２の確実復旧機能でもＪａｖａＶＭ及びＯＳの正常起動が確認されているので、ブートステータスレジスタ２９のＢＯＯＴフラグは反転しておらず、起動プログラム決定部４８は直前に動作していたＪａｖａＶＭ及びＯＳ（第１の確実復旧機能の場合には、ダウンロードされたＪａｖａＶＭ及びＯＳ、第２の確実復旧機能の場合には、ダウンロードする前のＪａｖａＶＭ及びＯＳ）を起動するように決定する（ステップＳ７７）。

[0075] 次に、図１７により正常解決機能及び復旧挑戦機能における起動アプリケーション決定部４９の処理を説明する。図１３のステップＳ６２でブー

トステータスレジスタ２９のＣＯＮＦＩＧフラグが反転された後、起動アプリケーション決定部４９はブートステータスレジスタ２９のＣＯＮＦＩＧフラグをチェックする（ステップＳ７９）。そして、起動アプリケーション決定部４９は、ＣＯＮＦＩＧフラグが反転しているか否か調べ（ステップＳ８０）、反転していれば、修復アプリケーションを起動するようにＣＯＮＦＩＧファイル（起動時に動作するプログラムを指定する定義が書かれたファイル）を書き換え（ステップＳ８１）、システムのＣＯＮＦＩＧを初期化してプログラムを実行し（ステップＳ８２）、ブートステータスレジスタ２９のＣＯＮＦＩＧフラグを反転させた（ステップＳ８３）後、システムをソフトウエアリセットさせる（ステップＳ８４）。ステップＳ８０でＣＯＮＦＩＧフラグが反転していなければ、起動アプリケーション決定部４９は、ＣＯＮＦＩＧフラグで設定されているアプリケーションを起動するように決定する（ステップＳ８５）。正常解決機能、復旧挑戦機能が動作している場合には、ＣＯＮＦＩＧフラグは反転していないので、起動アプリケーション決定部４９は、ＣＯＮＦＩＧフラグで設定されているアプリケーションを起動するように決定する（ステップＳ８５）。

[0076] また、図１８は確実復旧機能における起動アプリケーション決定部４９の処理を説明している。図１３のステップＳ６２でブートステータスレジスタ２９のＣＯＮＦＩＧフラグが反転された後、起動アプリケーション決定部４９はブートステータスレジスタ２９のＣＯＮＦＩＧフラグをチェックする（ステップＳ７９）。そして、起動アプリケーション決定部４９は、ＣＯＮＦＩＧフラグが反転しているか否か調べ（ステップＳ８０）、反転していれば、修復アプリケーションを起動するようにＣＯＮＦＩＧファイルを書き換え（ステップＳ８１）、ＣＯＮＦＩＧ初期化プログラムを実行し（ステップＳ８２）、ブートステータスレジスタ２９のＣＯＮＦＩＧフラグを反転させた（ステップＳ８３）後、システムをソフトウエアリセットさせる（ステ

ップＳ８４）。ステップＳ８０でＣＯＮＦＩＧフラグが反転していなければ、起動アプリケーション決定部４９は、ＣＯＮＦＩＧフラグで設定されているアプリケーションを起動するように決定する（ステップＳ８５）。第１又は第２の確実復旧機能が動作している場合には、正常解決機能、復旧挑戦機能でＪａｖａＶＭ＋ＯＳの正常動作が確認されているが、アプリケーション動作監視部５０でウォッチドックタイマ２８がタイムアウトしたため、ＣＯＮＦＩＧフラグが反転しているので、起動アプリケーション決定部４９は、修復アプリケーション２３を起動するようにＣＯＮＦＩＧファイルを書き換え（ステップＳ８１）、ＣＯＮＦＩＧフラグ初期化プログラムを実行し（ステップＳ８２）、ブートステータスレジスタ２９のＣＯＮＦＩＧフラグを反転させる（ステップＳ８３）。これにより、図１２のステップＳ５８で反転されたＣＯＮＦ１Ｇフラグはもとに戻る。さらに、起動アプリケーション決定部４９は、システムにソフトウエアリセットを掛ける（ステップＳ８４）。ソフトウエアリセットを掛けて、再度このステップへ来た時には、ＣＯＮＦＩＧフラグはもとに戻っているので、ＣＯＮＦＩＧファイルで設定されているアプリケーション（修復アプリケーション２３）を起動するように決定される。

[0077] 次に、図１９により正常解決機能、復旧挑戦機能、確実復旧機能におけるアプリケーション動作監視部５０の処理を説明する。図１３のステップＳ６３でウォッチドックタイマ２８がリセットされた（ステップＳ８６）後、アプリケーション動作監視部５０は通常処理を行ない（ステップＳ８７）、修復アプリケーション２３が起動しているか否かチェックする（ステップＳ８８）。修復アプリケーション２３が起動していない場合には、ウォッチドックタイマ２８がタイムアウトしているかチェックするが、修復アプリケーション２３が起動している場合には、ウォッチドックタイマ２８をチェックせず、ステップＳ８６の処理へ戻ってウォッチドックタイマ２８をリセット

する。従って、確実復旧機能が実行されている場合には、修復アプリケーション２３が起動しているので、決してウォッチドックタイマ２８がタイムアウトしない。これに対し、実行アプリケーション２２が実行されている正常解決機能または復旧挑戦機能の場合には、アプリケーション動作監視部５０はウォッチドックタイマ２８のタイムアウトをチェックし（ステップＳ８９）、ウォッチドックタイマ２８がタイムアウトならソフトウエアリセットを掛けて（ステップＳ９０）確実復旧機能へ移行し、タイムアウトしていなければ、ステップＳ８６の処理へ戻ってウォッチドックタイマ２８をリセットする。

[0078] しかして、ダウンロードしたＪａｖａＶＭ＋ＯＳ（新動作プログラム）が正常に起動し、実行アプリケーション２２も正常に起動した場合（正常解決機能の場合）の処理は、図１４、図１７、図１９の流れになる。また、ダウンロードしたＪａｖａＶＭ＋ＯＳ（新動作プログラム）が正常に起動せず、ダウンロードする前のＪａｖａＶＭ＋ＯＳ（旧動作プログラム）で正常に復旧した場合（復旧挑戦機能の場合）の処理は、図１５、図１７、図１９の流れになる。また、正常解決機能において、ダウンロードしたＪａｖａＶＭ＋ＯＳ（新動作プログラム）は正常に起動したが、実行アプリケーション２２が正常に動作せず、修復アプリケーション２３で確実に復旧した場合（第１の確実復旧機能の場合）の処理は、図１６、図１８、図１９の流れになる。また、確実復旧機能において、ダウンロードする前のＪａｖａＶＭ＋ＯＳ（旧動作プログラム）で実行アプリケーション２２が正常に動作せず、修復アプリケーション２３で確実に復旧した場合（第２の確実復旧機能の場合）の処理は、図１６、図１８、図１９の流れになる。そして、正常解決機能、復旧挑戦機能、確実復旧機能の３段階を段階的に行なうことで、本発明の目的が実現される。

[0079] 次に、正常解決機能におけるソフトウエアの流れを図２０に示す。この図に示すように、正常解決機能の場合には、ダウンロード部５１によって新しいＪａｖａ及びＯＳ（新動作プログラム２４）がダウンロードされ、現在動作していないＪａｖａVM及びＯＳ（旧動作プログラム２１）を新しいＪａｖａVM及びＯＳに置き換える。ついで、ブートプログラムを起動する。ＪａｖａVM＋ＯＳ起動／動作監視部５２は、起動するＪａｖａVM及びＯＳを決定し、壊れていないか、フリーズしないかをチェックし、フリーズすれば、復旧挑戦機能を実行する。また、アプリケーション起動／動作監視部５３は、実行アプリケーション２２を起動し、フリーズしないかチェックし、フリーズしたらソフトウエアリセットを掛け、第１の確実復旧機能を実行する。

[0080] また、復旧挑戦機能におけるソフトウエアの流れを図２１に示す。新しいＪａｖａVM及びＯＳの起動時にフリーズし、ソフトウエアリセットが掛かると、ＪａｖａVM＋ＯＳ起動／動作監視部５２は、ダウンロードする前に動作していた古いＪａｖａVM及びＯＳ（旧動作プログラム）を起動し直す。ついで、アプリケーション起動／動作監視部５３は、実行アプリケーション２２を起動し、フリーズしないかチェックする。そして、フリーズしたら、ソフトウエアリセットを掛け、第２の確実復旧機能を実行する。

[0081] また、第１の確実復旧機能におけるソフトウエアの流れを図２２に示す。アプリケーション起動／動作監視部５３は実行アプリケーション２２を監視しており、新しいＪａｖａVM及びＯＳで実行アプリケーション動作時にフリーズし、ソフトウエアリセットがかかると、新しいＪａｖａVM及びＯＳは正常に起動することを確認しているので、ＪａｖａVM＋ＯＳ起動／動作監視部５２により新しいＪａｖａVM及びＯＳを起動し直す。ついで、

アプリケーション起動／動作監視部５３が修復アプリケーションを起動し、復旧が完了する。

[0082] また、第２の確実復旧機能におけるソフトウエアの流れを図２３に示す。アプリケーション起動／動作監視部５３は実行アプリケーション２２を監視しており、古いＪａｖａVM及びＯＳで実行アプリケーションが動作時にフリーズし、ソフトウエアリセットがかかると、古いＪａｖａVM及びＯＳは正常に起動することを確認されているので、ＪａｖａVM＋ＯＳ起動／動作監視部５２により古いＪａｖａVM及びＯＳを起動し直す。ついで、アプリケーション起動／動作監視部５３は、修復アプリケーション２３を起動し、復旧を完了する。

[0083] 図２４は自動検針システム（データ収集器）６１のモジュール構成図である。この自動検針システム６１は、ＰＨＳ６２を介して家庭や管理センタと定期的にデータ通信を行なっている。そして、図２４に示すアップデート可能部分６３が管理センタからのデータによってアップデートされるようになっている。図２５は、図２４の自動検針システム６１のハードウエア構成図である。ＰＨＳ６２はＰＰＰドライバやシリアルドライバを介してシリアルポート６４と接続されている。ＲＯＭ６５には、起動プログラムやブートプログラム、ＣＯＮＦＩＧファイルが格納される。また、外部記憶装置であるコンパクトフラッシュメモリ（ＣＦ）６６には、自動検針用アプリケーション７０、修復アプリケーション７１、自動検針により得られたデータなどが格納される。図２６は、図２５のＲＯＭ６５とコンパクトフラッシュメモリ６６の状態を示している。図２６に示すＲＯＭ６５のＲＯＭ領域６７で、起動プログラムMが立ち上がり、自動検針用アプリケーションが動作している。図２４のアップデート可能部分６３において、バグフィックスを行なったり、ソフト的な機能追加やハード（機器）の追加によるドライバの追加を

行なった場合には、起動プログラムMの対応する個所を修正し、コンパイルする。修正、コンパイル済みの起動プログラムをM'とすると、起動プログラムM'をネットワークを通じて、図２６のＲＯＭ６５のＲＯＭ領域６８へ格納し、ＲＯＭ領域６８から起動し直す。また、起動プログラムM'の起動に失敗した場合には、ＲＯＭ領域６７のＭで起動し直す。また、ＣＯＮＦＩＧファイル６９のデータの破壊により自動検針用アプリケーション７０の起動に失敗した場合には、図２６のコンパクトフラッシュメモリ６６に格納している修復アプリケーション７１でデータの修復を行ない、必要最低限の機能を持ったアプリケーションで起動し直す。これにより、自動でアップデートを行え、かつ決してフリーズしないアップデートが完了する。なお、修復アプリケーションは上記に説明した、起動の設定を変更する機能のほか、ｔｅｌｎｅｔ、ＦＴＰなどの最低限の通信機能を確保するアプリケーションであっても良いし、複数の修復アプリケーションがあっても良い。状況に応じてＢＯＯＴフラグで必要な修復アプリケーションを立ち上げるようにするとよい。たとえば、修復アプリケーションに通信機能を持つことにより、ほとんど全ての機能がフリーズした後においても、通信機能をつかって遠隔操作でフリーズを解消し、アップデートでのフリーズを防止できる。